AFRL-VA-WP-TP-2003-331

# PATH ELONGATION FOR UAV TASK ASSIGNMENT

**Corey Schumacher**
**Phillip R. Chandler**
**Steven J. Rasmussen**
**David Walker**

**SEPTEMBER 2003**

**Approved for public release; distribution is unlimited.**



**AIR VEHICLES DIRECTORATE**
**AIR FORCE RESEARCH LABORATORY**
**AIR FORCE MATERIEL COMMAND**
**WRIGHT-PATTERSON AIR FORCE BASE, OH 45433-7542**

# REPORT DOCUMENTATION PAGE

*Form Approved OMB No. 0704-0188*



**1. REPORT DATE** *(DD-MM-YY)*
September 2003

**2. REPORT TYPE**
Conference Paper Preprint

**3. DATES COVERED** *(From - To)*

**4. TITLE AND SUBTITLE**
PATH ELONGATION FOR UAV TASK ASSIGNMENT

**5a. CONTRACT NUMBER**
In-house

**5b. GRANT NUMBER**

**5c. PROGRAM ELEMENT NUMBER**
N/A

**6. AUTHOR(S)**
Corey Schumacher and Phillip R. Chandler (AFRL/VACA)
Steven J. Rasmussen (Veridian, Inc.)
David Walker (Brigham Young University)

**5d. PROJECT NUMBER**
N/A

**5e. TASK NUMBER**
N/A

**5f. WORK UNIT NUMBER**
N/A

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
Control Theory Optimization Branch (AFRL/VACA)
Control Sciences Division
Air Vehicles Directorate
Air Force Research Laboratory, Air Force Materiel Command
Wright-Patterson AFB, OH 45433-7542

Veridian, Inc.

Mechanical Engineering Department
Brigham Young University
Provo, UT 84602

**8. PERFORMING ORGANIZATION REPORT NUMBER**
AFRL-VA-WP-TP-2003-331

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**
Air Vehicles Directorate
Air Force Research Laboratory
Air Force Materiel Command
Wright-Patterson Air Force Base, OH 45433-7542

**10. SPONSORING/MONITORING AGENCY ACRONYM(S)**
AFRL/VACD

**11. SPONSORING/MONITORING AGENCY REPORT NUMBER(S)**
AFRL-VA-WP-TP-2003-331

**12. DISTRIBUTION/AVAILABILITY STATEMENT**
Approved for public release; distribution is unlimited.

**13. SUPPLEMENTARY NOTES**
To be presented at the AIAA Guidance Navigation and Control Conference, Austin, TX, August 11-14, 2003. This material is declared a work of the U.S. Government and is not subject to copyright protection in the United States.

**14. ABSTRACT**
This paper presents a path planning and path-elongation method for wide area search munitions, a type of small Unmanned Air Vehicle. Variable-length paths are necessary to allow sufficient flexibility for efficient task assignment meeting timing constraints for UAVs. Five specific cases are developed that identify the best method of elongation based only on the initial position and heading of the vehicle with respect to the target. The cases include one linear direct method of path elongation and four cases which use an iterative approach for the nonlinear path elongation.

**15. SUBJECT TERMS**

| 16. SECURITY CLASSIFICATION OF: a. REPORT | b. ABSTRACT | c. THIS PAGE | 17. LIMITATION OF ABSTRACT: | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON (Monitor) |
|---|---|---|---|---|---|
| Unclassified | Unclassified | Unclassified | SAR | 24 | Edwin Allen<br>**19b. TELEPHONE NUMBER** *(Include Area Code)*<br>(937) 904-6521 |

**Standard Form 298 (Rev. 8-98)**
Prescribed by ANSI Std. Z39-18

# Path Elongation for UAV Task Assignment


Corey Schumacher*, Phillip R. Chandler*
Flight Control Division
Air Force Research Laboratory
(AFRL/VACA)
Wright-Patterson AFB, OH 45433-7531

Steven J. Rasmussen*
Veridian Inc.
Wright-Patterson AFB, OH

David Walker
Mechanical Engineering Department
Brigham Young University
Provo, Utah 84602



**Abstract**

This paper presents a path planning and path-elongation method for wide area search munitions, a type of small Unmanned Air Vehicle. Variable-length paths are necessary to allow sufficient flexibility for efficient task assignment meeting timing constraints for UAVs. Five specific cases are developed that identify the best method of elongation based only on the initial position and heading of the vehicle with respect to the target. The cases include one linear direct method of path elongation and four cases which use an iterative approach for the nonlinear path elongation. The iterative methods are similar to a Newton-Raphson search over a function for a specific value (a path of the desired length). The functions searched are the path length vs. the delay used, are monotonically increasing, and are very well behaved. The result is fast convergence to a small window around the desired value (path length). Multiple aspects of the path elongation program are presented. First, the problem setup is reviewed which includes the correction to the heading algorithm and the definition of the different cases and their associated elongation methods. The process of the iterative method is described and the results of the individual methods are presented. Results are shown for elongated paths for each of the five cases, and simulation results are shown using the path elongation algorithms in a task assignment problem.


## 1 Introduction

Task assignment for wide area search munitions can be effectively performed using an iterative network flow optimization [1,2]. One of the primary drivers of the complexity in this task assignment problem is the coupling between path planning and task assignment [3-5]. Many methods require that the path planning and task assignment elements of the overall problem be solved separately. In [2], it is shown that the ability to modify previously-computed paths can substantially improve the performance of the overall assignment algorithm. However, due to space constraints, many details about the path-elongation algorithms used in [2] are not given. That information is presented here. Further information on task assignment using network flows can be found in [6-8].

### 1.1 The Problem of Infeasibility

For this UAV problem, the vehicles perform three distinct duties on each of the targets. Once a target has been found during wide area search, the vehicles must classify the target, attack the target (if classified as a target of sufficient value), and then verify that the target was killed. Each vehicle in a scenario calculates the shortest possible path for it to complete each of the unassigned and incomplete tasks. The path lengths, which directly correspond to ETA for the constant-velocity vehicles, are then used to assign all the unassigned tasks to the available vehicles. Without consideration of timing, each target/task combination is assigned to an available vehicle that can accomplish it using the shortest available path. Whenever all available vehicles have paths that will arrive at a target subsequent task before the previous one will be completed, the paths are too short, the timing is infeasible, and the assignment cannot be made to any vehicle. Each target, in theory, will eventually be assigned and completely prosecuted when assignments are reallocated and there is a vehicle whose computed path to complete the next unassigned task is no longer infeasible. The results of this method are sub-optimal because the shortest, feasible path computed at a later time may actually arrive much later than the target could have been serviced if timing issues were considered. Therefore, the problem of infeasibility is that the allocation may fail to assign vehicles to targets due to constraints on timing coordination of multiple vehicles. This is especially true of multiple task tours, as shown in [1].

### 1.2 Initial Considerations

Obtaining paths of specific lengths can be accomplished in multiple ways. We currently only consider the elongation of "attack" and "verification" tasks, in which the final heading is not specified, and path elongation can be performed with no specific desire to maintain any given final heading. This means that a vehicle may elongate its path in a way that will change the final heading with which the vehicle approaches the task/target. We allow the final heading to change in an attempt to reduce the magnitude and complexity of the elongation problem. The result is that the final heading is primarily determined by the initial position of the vehicle and the desired time to service the target. An important element of the problem is that the vehicles under consideration can only fly at constant speed and altitude. Simply changing flight speed to modify arrival time is not an option. The path length must be modified to achieve the desired arrival time. The path-elongation methods presented in this paper have been developed based on a paradigm that encourages the use of straight and level flight whenever possible, because the vehicle's seeker is not gimbaled and cannot be used during turns.

*Member, AIAA

The creation of paths subject to timing constraints is a practical tool that can be used in many aspects of coordination control research. In the current allocation algorithms [2], vehicles are assigned to targets based on their shortest path. If the vehicle would arrive too soon it is assigned zero benefit to accomplish the task. It may be possible, rather than assign zero benefit, to maintain the assignment, generate a new trajectory that is feasible, and then continue the allocation. Overall, the total allocation may be more optimal. It is to this end that the path elongation functions are derived herein.

## 2 Problem Setup

Path elongation can be performed in a number of different ways. Different methods of elongation are needed depending on the position and heading of the vehicle relative to the target at the beginning of the trajectory. These initial conditions determine how the shortest trajectory will be computed, and will also determine which method of elongation will work best

### 2.1 Definition of Cases

The initial conditions define five different situations, or cases, that define the minimum distance trajectory and the best elongation method. Each case calls for a specific method of elongation or some degree of special treatment. Some of the cases are very similar to others, but are distinct enough in the details to require specific, individual treatment.

Without loss of generality, the cases are defined with the initial vehicle velocity at a heading of zero degrees (going from left to right), with the target at the origin. For any given initial position and heading, with any arbitrary final target position, the coordinates are transformed to the zero initial heading with the target at the origin for classification within one of the five cases. The boundaries of the individual case types in the transformed coordinates are shown in Figure 1. The case the vehicle is found to be in determines the method of elongation to be used and the window of all possible completion times for the given task.

The boundaries in the figure represent mathematical case limits for the vehicle relative to the goal, assuming the vehicle travels from left to right after the coordinate transformation. The limits on vehicle position for each case are primarily dictated by the position of the left and right turning circles (defined by the minimum turn radius) for the vehicle. The distance from the target to these circle centers is needed in determining how the path can be elongated, and where path length discontinuities will occur during path elongation. The discontinuities arise when the inside turn circle center is sufficiently far from the goal at the vehicle's initial position, but then becomes too close to the goal during elongation; that is, when the inside turn circle is outside radial distance "b" initially (the distances represented are defined below), but crosses below this radial distance during the delay.

The thin green circles in the Figure 1 represent the limiting distances for the turning circle center distances, and are labeled from a to d. These radial distances from the target represent limitations on the vehicle such as minimum turning radius and required stand off distance for task completion. The radial distances shown are summarized as:

- "a" – Sensor Stand Off Distance, R; this is the distance the vehicle must be from the target in order for it's sensor to pass over the target.
- "b" – The minimum radial distance from target to final turn circle center for vehicle position to satisfy sensor stand off limit boundaries. This distance is given as $L = \sqrt{R^2 + r^2}$, for $r = TurnRadius$.
- "c" – This is a distance of $L + r$, and is used in defining Case *I* and the transitions and limits for Case *III*.
- "d" – A distance of $L + 2r$; This distance is a boundary for the outer turn circle, and determines whether the vehicle is in Case *III* or *IV*, and whether or not there will be discontinuities in the path length elongation.

The radial distances described above are used to implement the mathematical boundaries shown in Figure 1 and determine which case applies. The boundaries are created, and the vehicle position is tested against them in the order the boundaries are numbered in the figure above. The boundary tests used are shown below from 1 to 8. In the statements below, "x" and "y" refer to the x and y coordinates of the vehicle in the transformed coordinate system. Similarly, "$rc_{out}$" and "$rc_{in}$" represent the radial distance from the goal to the outer and inner initial turn circle centers respectively. The need for the boundaries set forth below becomes clear when considering how to best elongate the path.

1. Test if $x < 0$, if true, continue through tests 2-5; else go to tests 6-8.
2. $|y| \geq (L + r)$, if true, vehicle is in Case *I* (where $L = \sqrt{R^2 + r^2}$).
3. If $rc_{out} \geq (L + 2r)$, then Case *III*.
4. If $rc_{out} \leq rc_{\pi}$, and $rc_{in} < L$, then Case *II*.
5. If $rc_{in} \geq L$, and $rc_{out} < (L + 2r)$, then Case *IV*.
   - Else test 1 is true but tests 2-5 are false and it is in Case *V*.
6. If $|y| \leq 2r$, then Case *II*.

7. If $|y| \le \sqrt{\left(rc_{\pi}{}^{2} - x^{2}\right)} - r$, then the vehicle is in Case *II*.
8. If $|y| < \sqrt{\left(L^{2} - x^{2}\right)} + r$, then Case *V*.
   - Else it is outside boundary 8 ($|y| > boundary8$), and is in Case *I*. (if it does not specifically meet any of the conditions above).

Now the problem has been fully classified. First, the final approach headings were generated that would have the shortest path length. The final heading obtained directly implies what the final path will look like, making it possible to know how to best elongate the flight trajectory. The cases have been defined to take advantage of the flight path that is obtained based only on the initial position and heading of the vehicle with respect to the target. With the vehicle position classified in this manner, the path is now ready for elongation.

## 3 Paths of Desired Lengths

The method of path elongation for each case will now be set forth and described. The case of the vehicle and the method of elongation will determine the total range of possible, feasible path lengths that can be obtained from the algorithm. This range of path lengths represents window of time in which it is possible for a vehicle to complete a given task. The cases, final heading corresponding to the shortest path, and all the elongated paths, are developed from the shortest possible path. Except for Case *II*, all elongation methods are iterative. Once the desired length of the path is determined, an actual path of the desired length is then obtained using case specific methods.

### 3.1 Case Specific Elongation Methods

The defined cases and their associated methods are closely tied together. The cases were defined first because it was more logical to the presentation, even though the cases and the methods were generated in parallel. The elongation methods for the specific cases involved either adding straight segments to the path where the shortest path would turn, or turning immediately but in the opposite direction of the first turn for the shortest path, or a combination of an opposite turn and a straight segment.

#### 3.1.1 Case *I*

A vehicle is in case *I* when it can elongate its path by any amount greater than zero by continuing straight on it's initial velocity heading. In this case the path length can be changed continuously, without any discrete jumps, to obtain any length between the shortest path length and infinity (with the only limit being the fuel of the vehicle). In Figure 2 below, the vehicle path shown on top is in case *I* and can elongate its path continuously. The vehicle path shown at the bottom of Figure 2 is too close to the target (distance perpendicular to velocity direction) to elongate continuously by continuing straight. An additional benefit obtained from continuing straight is that the path is elongated while also being able to continue searching for more targets along the vehicle's current path.

#### 3.1.2 Case *II*

Case *II* is the only case in which a desired path length can be obtained directly. The only condition for a vehicle to be in case *II* is that the vehicle must turn at least 180° in one direction. Whenever this is the case, a path extension equal to half the desired elongation distance can be added to the path on both sides of the 180° turn. This works even if the turn is completed through multiple waypoints. In the figure below the elongated paths are in solid lines and the original paths are shown in dashed lines. The case *II* elongation method has two attractive elongation characteristics due to the fact that the elongation occurs in the middle of the path. First, the final heading is unchanged, so subsequent tasks will not need to re-plan their routes based on the initial velocity direction for that task. The second beneficial attribute of the 180° turn path elongation is that, just like in Case *I*, the vehicle can create and follow paths of any length greater than or equal to that of the minimum path. Figure 3 demonstrates how the path elongation is accomplished in Case *II*.

#### 3.1.3 Case *III*

Cases *III* and *IV* are very similar. The only difference between them is that case *IV* (shown in Figure 4) is close enough to the target to cause discontinuities in the path length during elongation. This means that a vehicle in case *III* can find an acceptable path of any length longer than the minimum, but a vehicle in case *IV* will have a range of possible path lengths that it is not possible for the vehicle to obtain.

The elongation method for cases *III* and *IV* is a two-part elongation involving both turning away and continuing straight (if the desired path length involves a large enough elongation of the shortest path). The length of the elongation will determine whether both methods are used, or if only the initial turning away. In both cases the vehicle requires only a single turn in the shortest length path. That is, the vehicle is far enough away from the target to turn directly towards it until it is facing the target. If the elongation involves both methods, a vehicle in case *III* will turn away until the vehicle's new position and heading fit under a case *I*. At this point the path elongation will switch to case *I* and iterate to find the elongated path by continuing straight. The point at which the vehicle transitions to case *I* is the critical point for the case *III* vehicle.

#### 3.1.4 Case *IV*

Case IV is shown in Figure 5. Path elongation for a vehicle in case *IV* more complicated due to the path

length discontinuities. The vehicle will perform in exactly the same manner as case *III* except when it is near/in the discontinuity. Due to the jump in path length there are two critical points in case *IV*, and two timing windows. The first critical point is where the discontinuity begins, and the path length associated with an elongation to this point is a bound on the upper value of the first timing window. If the desired path length is before the first critical point, then the algorithm iterates on an elongation between zero the first critical point to find the path. The second critical point is where the discontinuity ends, and the path associated with it is the lower bound on the second timing window. Once the vehicle enters too close to the target, the best course is to continue turning away until it reaches the second critical point. If the desired path length is in the discontinuity (and therefore infeasible), then the path through the second critical point is returned as the best feasible value. If the path is still not long enough after the elongation through the second critical point, the vehicle transitions to case *I* and iterates on a straight line elongation to find a path of the desired length.

#### 3.1.5 Case *V*

Case *V*, shown in Figure 6, is basically a special instance of case *I*. When a vehicle is in case *V*, the shortest path to the goal always involves an immediate turn away from the goal. If the path was elongated in the same way as case *I* (by continuing search), the same algorithms do not apply without large modifications. These modifications require a different approach in the algorithms, making the code less general. The solution used to resolve this problem was to allow the vehicle to maintain the same course through the first turn in the opposite direction. Once the vehicle has completed this turn it is outside the complications, allowing case *I* functions to apply directly. At this point the vehicle begins to perform elongation by continuing straight. This method, like the nearly identical method for case *I*, produces a continuous range of possible path lengths. Again, the upper bound on the path length is constrained by the fuel limits of the vehicle. The resulting second turn in the path is always nearly, but not quite, 180°. This prevents the case from switching to case *II*, rather than case *I*. However, the large turn makes the iteration for case *I* work very quickly, requiring only two or three iterations to converge.

### 3.2 Iterative Path Elongation

It has been explained that only one of the five cases results in a truly linear path elongation and can be solved directly. The other cases result in nonlinear elongations and direct solutions for a path of a specified length could not be found. To find paths for these cases, we use an iterative method much like a numerical Newton-Raphson search.

The "functions" that the Newton-Raphson-like method is working on depend on the case in which the vehicle is. Different cases have paths that are more nonlinear than others. Also, the domain of the "function" varies since in some cases the elongation is obtained through flying straight, and in other cases it is obtained by turning in the opposite direction. The range of the functions is always the resulting path length (or can be measured in ETA since velocity and time are equivalent for the constant velocity vehicles). The domain is the amount the vehicle delays its approach to the target through the elongation method. In case *I*, where elongation is achieved by flying straight, the domain is a distance, in feet, that the vehicle flies before turning toward the target. In cases *III* and *IV* (when the desired path length is found between the initial path and the first critical point), the "x" value of the function is the delay angle, in radians, that the vehicle turns away before turning back towards the target. In every case, the domain will always be positive since the initial vehicle position is the point of zero elongation. A negative elongation is not possible, as we are starting from the shortest achievable path to the target.

The desired path length will always need to have a window of possible values for the iteration. If an exact value is required the number of iterations to find that path will go to infinity. As a result, a window of possible values will be needed to ensure that a suitable path is found in a reasonable number of iterations. In the present work a window was hard coded into the program at 0.05%. For example, if a path of 15,000 feet is desired, an acceptable path will be between 15,000 and 15,000*1.0005 = 15007.5 feet.

The iteration begins by generating a new path and determining its length given some initial elongation. The initial elongation value for cases *I* and *V* (where elongation is a straight flight) is 55% of the total desired length of elongation. Cases *III* and *IV* have a natural initial elongation equal to a turn to the first critical point. This is because if the path is longer than the path created at the critical point the iterative method must change to a different case. If the path is too long when evaluated for an elongation at the critical point, then it serves as an initial elongation for the iteration.

The iteration begins by linearly connecting the last two computed points in the function. A point is a path length and delay distance pair. The line connecting the points linearly estimates the needed value of the elongation to get the desired path length. The path for the estimated elongation is then created and the new path length is compared to that of the desired length. If the path length is not within the acceptable window, the iteration continues by using the last two points calculated in the iteration. The process is illustrated in Figure 7. In the figure, only three additional paths were computed before the third path was found within the acceptable window of path lengths. When the last two points computed produce an estimated elongation that is infeasible, the two closest points that have path lengths that window the desired length are used instead of the last two computed points. The estimated delay is

infeasible when it is estimated to be negative, or when iterating on the initial opposite turn for cases III and IV and the estimated turn delay angle is greater than the first critical point. An infeasible estimated delay can occur due to nonlinear effects in the function.

## 4 Path Elongation Results and Discussion

The iteration used, as described above, proves to be very fast. All elongated paths are generated off the original, shortest path. Using the shortest path allows the critical trajectory information to be known, such as initial and final turn directions, before the elongated path is computed. This simplification results in a single path being computed per iteration rather than the four computed in order to obtain the original minimum distance trajectory. By using the information from the original path, the algorithm requires less computation per iteration and increases the speed with which elongated paths are returned. The speed of the algorithm can be shown by the number of elongated paths that can be generated in a given amount of time. In testing the algorithm, elongated trajectories were generated for every case. Holding the target at the origin, the algorithm was tested in a loop with the initial vehicle position at every node in a discrete grid of the space surrounding the target. The tests performed produced 200+ elongated paths per second.

Examples of an elongated path for each Case are shown in Figures 8-12. In each figure, the initial position of the vehicle is marked by a blue 0. The initial vehicle path is shown in green, as are turn circles used in calculating vehicle paths. The elongated vehicle path is shown in blue. In all cases the target is located at x = 8900 ft, y = -4100 ft. All of the paths presented are for verification tasks with a sensor standoff distance of 5100 ft. This is important, because the task is completed when the vehicle reaches this distance from the target with a straight heading toward the target. This means that only the portion of the path shown *before* the sensor standoff is reached is elongated in the examples given. In every case, a path of the desired patch length is achieved. The new path length (and task completion time) are related to the initial path length (completion time) by 1/(scale factor). However, in Case IV, there is a discontinuity, and a path only a little shorter than that shown would not be feasible, due to the turn constraints of the vehicle.

## 5 Assignment Methodology

The iterative network flow task assignment methodology described in [2] has been implemented in our multi-vehicle, multi-target coordinated-control simulation with the variable-length path-planning methods discussed here.

The assignment algorithm is based on a network flow optimization, shown in Figure 13. There are N sources, one for each vehicle, a set of tasks corresponding to the present state of each known target, and a sink. Each source must flow to the sink, either by flowing through a target node, or directly via the "search" path. All flow values $x_{ij}$ are constrained to be either 0 or 1. These flows $x_{ij}$ correspond with vehicle-target task assignments. Each unit of flow along an arc (task assignment) has a "benefit" which is an expected future value. The optimal solution maximizes total value.

The network optimization model can be expressed as:

$$\max J = \sum_{i,j} c_{ij} x_{ij} \qquad (1)$$

Subject to:

$$\sum_{j,s} \left( x_{ij} + x_{is} \right) = 1, \qquad , \forall i = 1, \ldots, n \qquad (2)$$

$$x_{j,k} - \sum_{i} x_{i,j} = 0, \qquad \forall j = 1, \ldots, m \qquad (3)$$

$$\sum_{i} x_{is} + \sum_{j} x_{jk} = n, \qquad , n = \# UAVs \qquad (4)$$

$$x \leq 1 \quad , \quad x \geq 0 \qquad (5), (6)$$

This particular model is a capacitated transshipment problem (CTP), a special case of a linear programming problem. The constraints ensure flow balance at each node, that every vehicle receives a task assignment, and that each task is only assigned once.

Due to the integrality property, it is not normally possible to simultaneously assign multiple vehicles to a single target, or multiple targets to a single vehicle. However, using the network assignment iteratively, "tours" of multiple assignments can be determined. This is done by solving the initial assignment problem, but only finalizing the assignment with the shortest ETA. The assignment problem is then updated assuming that assignment is performed, updating target and vehicle states, and running the assignment again. This iteration can be repeated until all of the vehicles have been assigned terminal attack tasks, or until all of the target assignments have been fully distributed. The target assignments are complete when classification, attack, and battle damage assessment tasks have been assigned for all known targets. Assignments must be recomputed if a new target is found or a munition fails to complete an assigned task.

A potential complication arises if we assume decoupling between path planning and task assignment. Minimum-time trajectories are calculated for each vehicle to perform each needed task, and these are then sent to the assignment algorithm and used in calculating the task benefits $c_{ij}$. If the minimum-time trajectory does not satisfy the timing constraints imposed by previous tasks, a new path must be calculated that will meet the timing constraints.

The weights $c_{ij}$ determine what actions the vehicles will take. They are based on the expected values of performing

each task. Killing a high-value target is assigned the most value, with other tasks generating less of a benefit. The weights are chosen so that vehicles with little remaining fuel will be assigned attack tasks, due to the loss of the vehicle inherent in attacking. Overall, the chosen weights tend to result in the least possible lost search time for the execution of a particular task. A "memory weighting factor" is used to prevent churning, which is when a vehicle's task is repeatedly changed as the assignment algorithm is re-run Full details of the iterative multiple assignment algorithm and weight selection with fixed-length paths can be found in [1]. A more detailed discussion of the incorporation of variable-length paths into the assignment algorithm is found in [2]. Theoretical foundation for network flows and task assignment can be found in [8].

**6 Assignment Results**

The scenario has eight Wide Area Search Munitions performing a search for targets in a rectangular area. The WASM are using a simple "mowing the grass" search pattern. There are up to 5 different target types possible in the simulation, including a "non-target" target type for objects that appear similar to targets but which may be distinguishable as non-targets by the ATR.

For the simulation results presented, eight vehicles are searching an area containing three targets. The search vehicles are initialized in a staggered row formation, with fifteen minutes of flight time remaining, out of a maximum thirty minutes. This assumes that the vehicles have been searching for fifteen minutes and then find a cluster of potential targets.

Figure 14 shows vehicle flight paths and target locations with minimum-length paths. The colored rectangles represent the sensor footprints of the searching vehicles, and the numbers are the target locations. Colored lines show flight paths. Targets are numbered 1,2,3. As soon as each target is discovered, classification, attack, and verification tasks are assigned for that target. Since the task allocation algorithm is performed each time a task is completed, it is possible for a vehicle's assignment to change based on new target information, although the memory weighting prevents this from happening if a potential new assignment is not a substantial improvement. In this case, the Classify and Attack tasks on target 2 are assigned to Vehicle 8, and they are performed after the verification tasks on Targets 1 and 3, which Vehicle 8 also performs. The assignment of this series of tasks to Vehicle 8 delays the classification and attack on Target 2 long enough that none of the minimum-length verification paths calculated by the other vehicles is long enough to meet the timing constraints. Accordingly, no vehicle verifies the destruction of target 2.

Figure 15 shows vehicle flight paths and target locations with variable-length paths. Whenever the minimum-length path does not satisfy the timing constraints, a new path that satisfies the constraints, and is near the minimum possible path length that satisfies the constraints, is calculated for each vehicle. This time, all of the tasks are completed. The assignments are identical to the minimum-length-path-only case, except that Vehicle 1 calculates an extended path that allows it to verify the destruction of Target 2 shortly after Target 2 is struck. Vehicle 1 extends its search path in a straight line and then comes back to Target 2 to perform verification just after it is attacked by Vehicle 8. This is an example of a Case I path extension. The variable-length path generation guarantees that all feasible tasks will be completed, if fuel constraints allow. Even with the iterative, variable-length path-planning, the overall assignment is still very fast, and can be implemented in real-time. A full assignment calculation takes only a few seconds on a 700 MHz processor, with much of the computation performed with unoptimized MatLab code.

**7 Conclusions and Future Work**

A method for elongating minimum-distance paths for a constant-velocity flight vehicle was presented. This method requires classifying the elongation problem into one of five "cases" based on the vehicle's initial position and heading relative to the target (or desired final waypoint). Slightly different path elongation techniques are required based on the initial case. The elongated paths can be used in a task assignment algorithm to prevent artificial infeasibility of assignments and improve the performance of the overall path planning and task assignment function. Simulation results were shown illustrating the use of the path elongation algorithms, and how they can improve performance in computing multiple-task tours using an iterative network flow task assignment algorithm.

References


1. Schumacher, C, Chandler, P. R, Rasmussen, S. J., "Task Allocation for Wide Area Search Munitions Via Iterative Network Flow", Proceedings of the 2002 AIAA Guidance, Navigation, and Control Conference.

2. Schumacher, C, Chandler, P. R, Rasmussen, S. J., "Task Allocation for Wide Area Search Munitions with Variable Path Length", Proceedings of the 2003 American Control Conference

3. "Cooperative Control: Models, Applications and Algorithms," Editor: Sergiy Butenko, Kluwer Academic Publishers, 2002.

4. Chandler, Phillip R., Pachter, Meir, "Hierarchical Control of Autonomous Teams," Proceedings of the

2001 AIAA Guidance, Navigation, and Control Conference

5. P. R. Chandler, S. Rasmussen, and M. Pachter, "UAV Cooperative Control," in Proceedings of the 2000 AIAA Guidance, Navigation, and Control Conference

6. Schumacher, C, Chandler, P.R, Rasmussen, S. R., "Task Allocation for Wide Area Search Munitions Via Network Flow Optimization", Proceedings of the 2001 AIAA Guidance, Navigation, and Control Conference.

7. Nygard, K. E., Chandler, P R. , Pachter, M.., "Dynamic Network Flow Optimization Models for Air Vehicle Resource Allocation," Proceedings of the Automatic Control Conference, June 2001.

8. Ford, L. R Jr. and D. R. Fulkerson, "Flows in Networks," Princeton University Press, Princeton, NJ, 1962

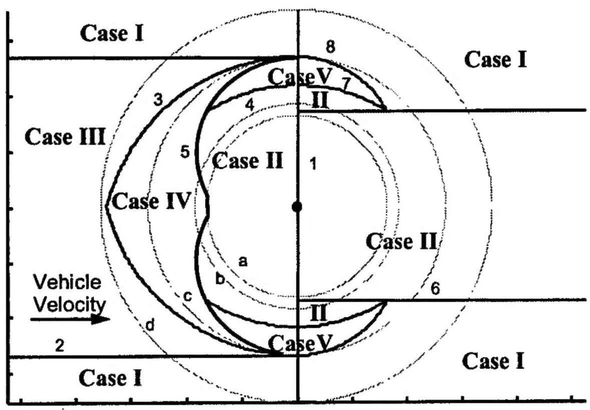


Figure 1 - The Figure shows the boundaries for the various cases. The separate boundaries are numbered in order of use, and are defined in the text. In green, and lettered from a to d are radial distances from the goal critical to the case construction.

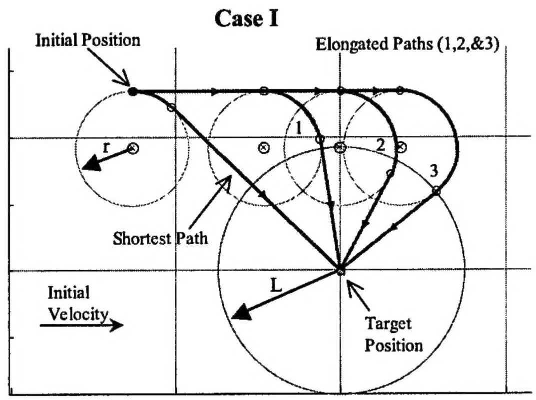


Figure 2 – The trajectory shown on top represents a vehicle in case *I* because it can be elongated indefinitely without path length discontinuities.

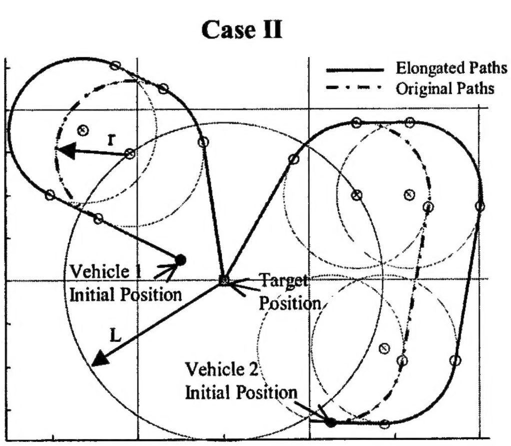


Figure 3 – Shown are two possible paths elongated using the case *II* 180° turn elongation method.

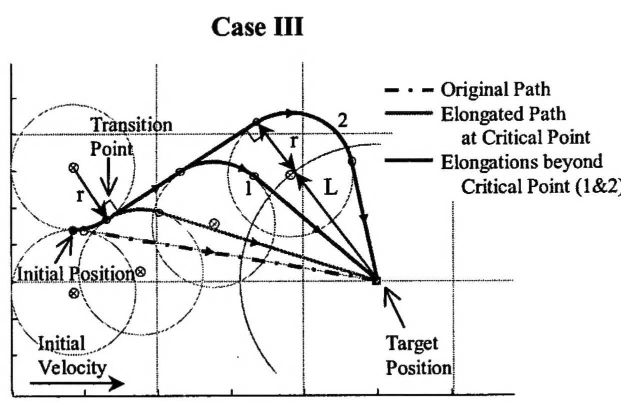


Figure 4 – The figure demonstrates the elongation method associated with case *III*, and shows three elongations.

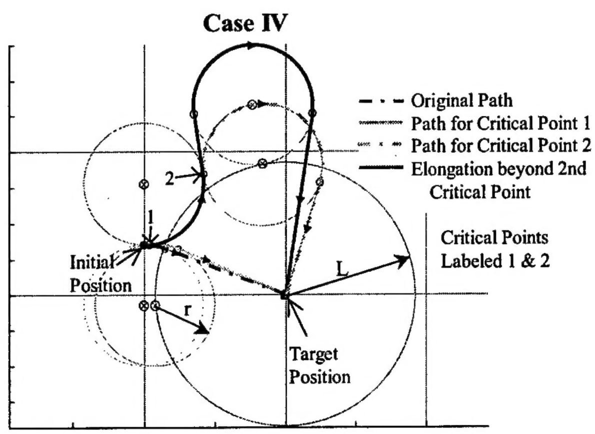


Figure 5 – The figure illustrates the original path, the paths for elongations to each critical point, and a path beyond the discontinuity for case *IV*.

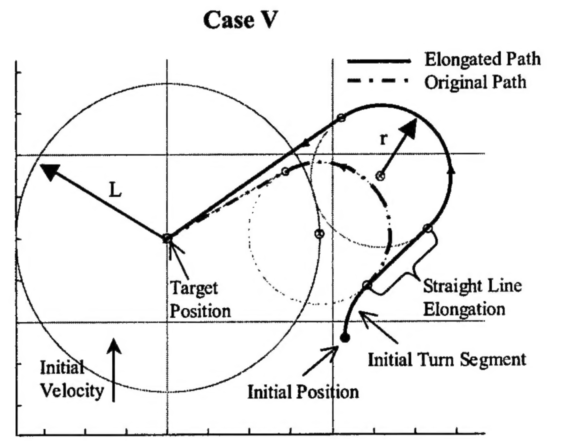


Figure 6 –Elongation for case *V* is performed by switching to case *I* after the initial first turn (which is maintained in the new path).

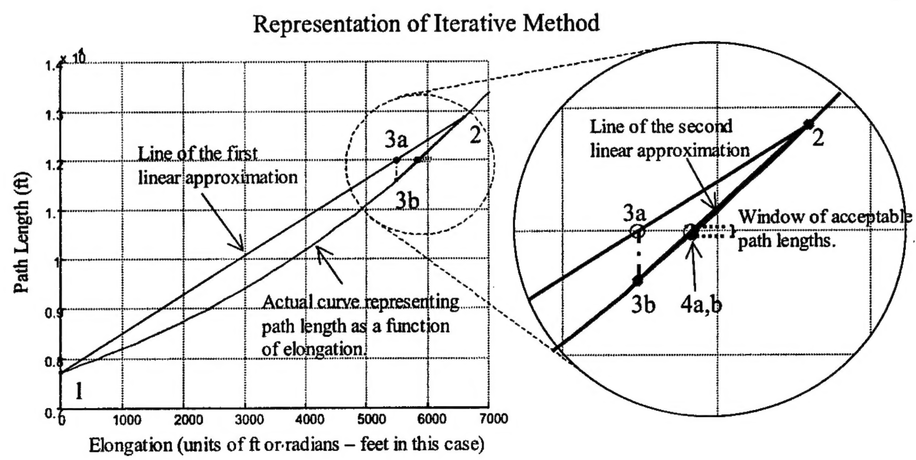


Figure 7 - The figure demonstrates the progress of the iterative method showing the four paths used to find the final path of a desired length. Path 1 is the original, shortest path. Path 2 is the first new path computed. The elongation for paths 3 and 4 are obtained from linear approximations based on previously computed paths. 3a and 4a are the expected path lengths for the elongation. 3b and 4b are the actual path lengths.

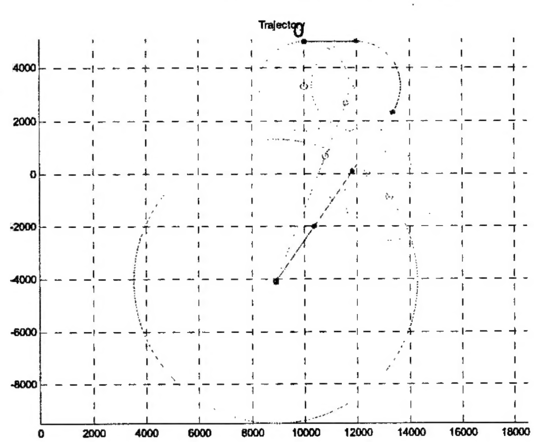


Figure 8: Elongation Example Case I. Scale Factor = 0.5

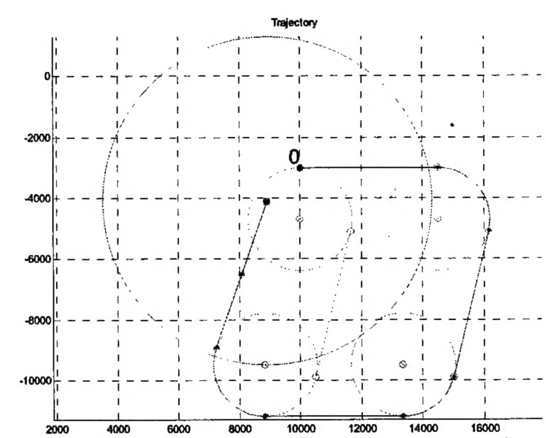


Figure 9: Elongation Example Case II. Scale Factor = 0.6

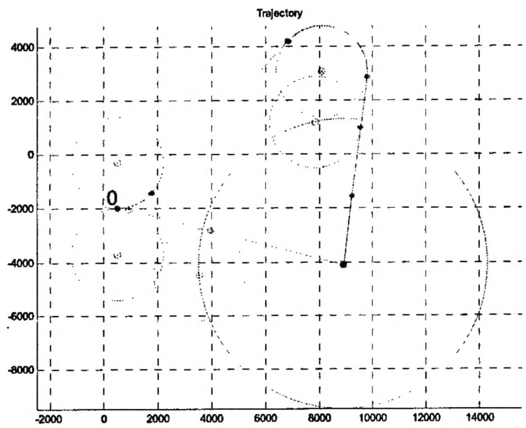


Figure 10: Elongation Example Case III. Scale Factor = 0.2

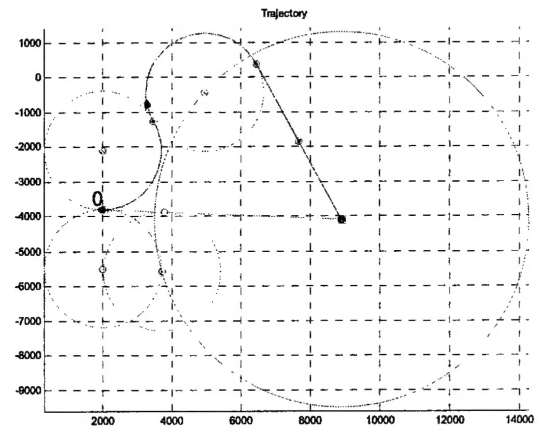


Figure 11: Elongation Example Case IV. Scale Factor = 0.4

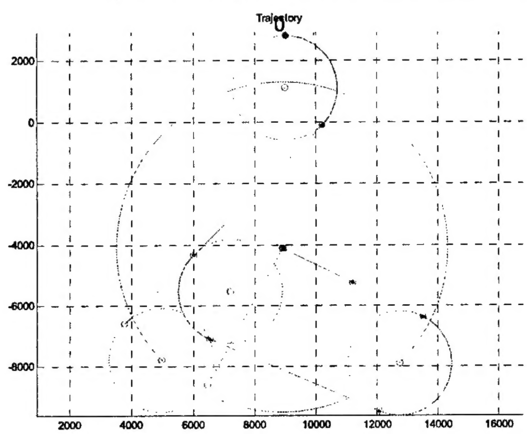


Figure 12: Elongation Example Case V. Scale Factor = 0.7

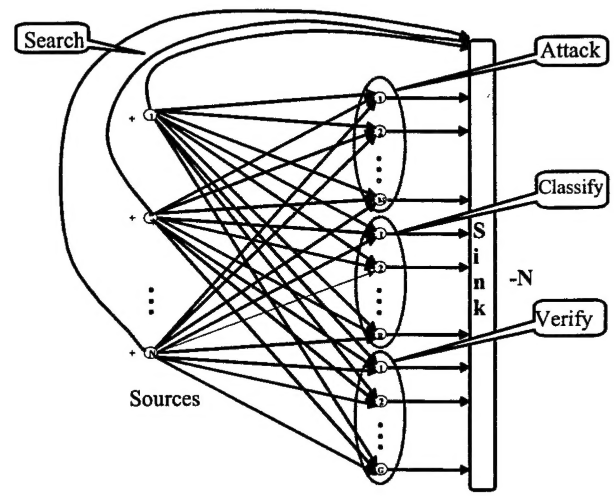


Figure 13: Network Flow Model for Task Allocation

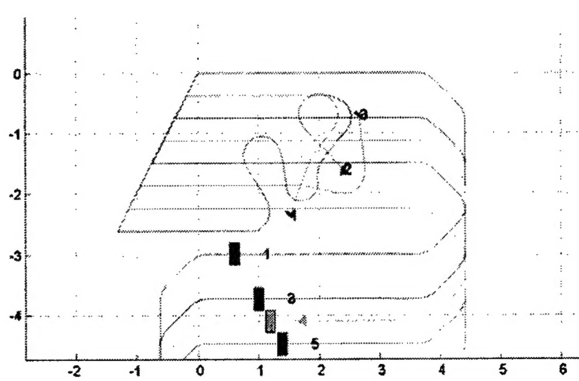

Figure 14: Vehicle Paths and Target Locations with Minimum Length Paths

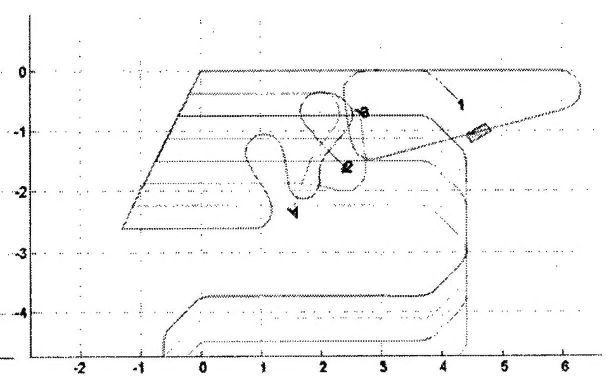

Figure 15: Vehicle Paths and Target Locations with Variable Length Paths